LETTERKENNY INSTITUTE OF TECHNOLOGY
ASSIGNMENT COVER SHEET

Lecturer's Name: Angela Sweeney

Assessment Title: Data Warehouse Reporting and Analytics Lab Report

Work to be submitted to: Angela Sweeney

Date for submission of work: October 22, 2023

Place and time for submitting work: ______

To be completed by the Student

Student's Name: Arya Sasi

Class: MSc Big Data Analytics

Subject/Module: Business intelligence

Word Count (where applicable): ______

I confirm that the work submitted has been produced solely through my own efforts.

Student's signature: ARYA SASI Date: October 22, 2023

Notes

Penalties: The total marks available for an assessment is reduced by 15% for work submitted up to one week late. The total marks available are reduced by 30% for work up to two weeks late. Assessment work received more than two weeks late will receive a mark of zero.

Continuous Assessment: For students repeating an examination, marks awarded for continuous assessment shall normally be carried forward from the original examination to the repeat examination.

Declaration:

I declare that this work is entirely my own and does not contain the words or ideas of someone else, whether published or not, without specific acknowledgement by relevant referencing. I have read and understood the LYIT Plagiarism Policy on the "Student & Academic Policies" section of the LYIT Website and understand plagiarism to include:

- Direct copying of text, images and other materials (electronic or otherwise) from a book, article, fellow student's essay, handout, web page or other source without proper acknowledgement.
- Claiming individual ideas derived from a book, article etc. as one's own and incorporating them into one's work without acknowledging the source of these ideas.
- Overly depending on the work of one or more other sources without proper acknowledgement of the source, by constructing an essay, project etc., extracting large sections of text from another source and merely linking these together with a few of one's own sentences.

I understand that it is my responsibility to familiarise myself with and to follow the Institute's Assessment Regulations. I acknowledge that Incidents of alleged plagiarism and cheating are dealt with in accordance with the Institute's Assessment Regulations and that penalties will be applied if I breach this policy.

Signed: ARYA SASI Date: October 22, 2023

Description

Scalar functions in SQL(Structured Query Language) are built-in or user-defined functions that accept one or more parameters and return a single value. Character data, such as strings, can be altered and transformed using scalar functions such as SQL character functions. SQL functions can be divided into two categories: scalar and aggregate.

This learning objective focuses on your capacity to use SQL functions for data transformation, cleaning, and classification. You may format data, conduct computations, and more using SQL functions to change data within your query. The SUM, COUNT, AVG, CASE, CONCAT, DATE, and other frequently used functions are only a few examples. You can efficiently use these features to prepare data for analysis and reporting.Performance is essential in a data warehousing scenario since there are often massive volumes of data involved.

Materialized views, which are also referred to as indexed views or summary tables, are precomputed views that hold aggregated or transformed data, obviating the need for intricate computations during query execution. You may greatly enhance query performance in data warehousing systems by generating and utilizing Materialized views.Performing elaborate analyses on data with analytical SQL entails employing sophisticated SQL constructs. This comprises strategies like window functions (for example, RANK, LEAD, and LAG), statistical functions, and more to draw conclusions from data. For data aggregation and summarization in data warehousing, the GROUP BY clause is crucial. In addition to ROLLUP, CUBE, GROUPING SETS, and analytical operations, GROUP BY has extensions that allow for multidimensional analysis and effective reporting.When working with massive datasets and data warehousing solutions, these abilities are crucial for data professionals.

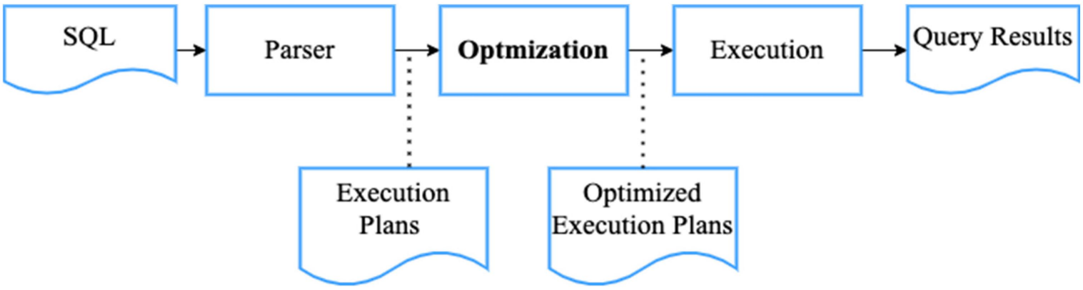


Figure - How SQL works in SQLDeveloper.

Objectives

1. Write a query to return sales total grouped by channel id and product id. (No SQL extensions to group by required here). Sample of the output is below.What is the Explain plan cost and table access?

2. Create a Materialized View called SALES_CHAN_PROD_MV based on the query you wrote above for Question 1.Show the Materialized View being used by query rewrite by noting the Explain plan cost and table access.Ensure that you describe the output and reasons for the explain plan cost reduction.

3. Write a query to return sales total grouped by channel description and product name. (No SQL extensions to GROUP BY required here). A sample of the output is as follows (225 rows):What is the Explain plan cost and table access?

4. Does the Query written in Question 3 above use the Materialized View SALES _CHAN _ PROD_MV?. If not explain why. Document your observations.

5. Create an appropriate Dimensional object to allow Q3 to use the Materialized View SALES _CHAN_PROD_MV.

6. Rerun Query 3. Does it use the materialized view SALES_CHAN_PROD_MV? What is the Explain plan cost and table access?

7. Write a query to produce a report for management that require sales total grouped by channel description, product category and country name showing a total at each level of aggregation for France and Italy. Management do not want Peripherals and Accessories, Hardware or Photo product categories included in the report. Improve the readability of the report by using decode(29 records are returned as you see in the report in Figure 3).Experiment with ROLLUP, PARTIAL ROLLUP, CUBE, GROUPING SETS AND GROUPING. Document your results and observations.

TASK 1

Objectives

Write a query to return sales total grouped by channel id and product id. (No SQL extensions to group by required here).

Method

First Step is to create the given tables such as Sales,Channels,Product.We already created the necessary tables to do these.After that we need to write the select Query to return the above mentioned Aim.

Query :

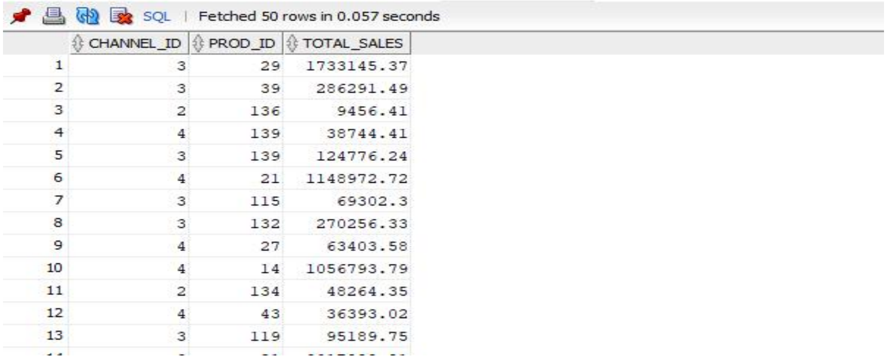
```
SELECT channel_id,prod_id,SUM(amount_sold) AS total_sales

FROM sales GROUP BY channel_id, prod_id;
```

In this query:

- sales is the placeholder for your actual table name where you store the sales data and are selecting the channel id and product id columns.
- Using the SUM() function to calculate the total sales amount for each combination of channel id and product id.
- Finally, grouping the results by channel id and product id using the GROUP BY clause.

The output of this query will give you the sales total grouped by channel id and product id, similar to the sample output you provided and the execution plan cost of sales table is 5.

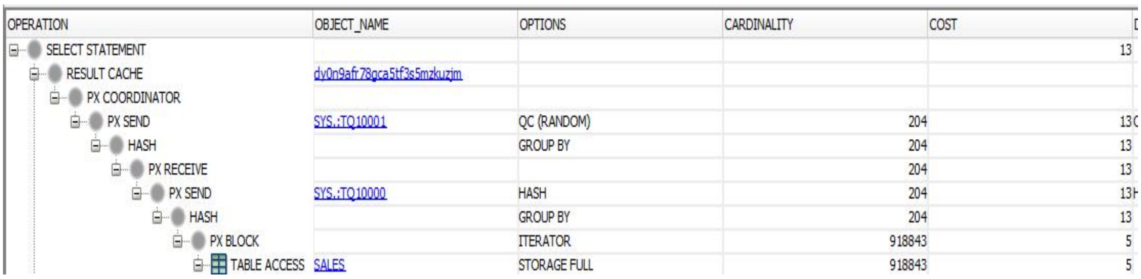
OUTPUT



Fetched 50 rows in 0.057 seconds

	CHANNEL_ID	PROD_ID	TOTAL_SALES
1	3	29	1733145.37
2	3	39	286291.49
3	2	136	9456.41
4	4	139	38744.41
5	3	139	124776.24
6	4	21	1148972.72
7	3	115	69302.3
8	3	132	270256.33
9	4	27	63403.58
10	4	14	1056793.79
11	2	134	48264.35
12	4	43	36393.02
13	3	119	95189.75

Figure 1 - Sales total grouped by channel_id and prod_id



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				13
RESULT CACHE	dy0n9afr78gca5tf3s5mzkuzjm			
PX COORDINATOR				
PX SEND	SYS.:TQ10001	QC (RANDOM)	204	13
HASH		GROUP BY	204	13
PX RECEIVE			204	13
PX SEND	SYS.:TQ10000	HASH	204	13
HASH		GROUP BY	204	13
PX BLOCK		ITERATOR	918843	5
TABLE ACCESS	SALES	STORAGE FULL	918843	5

Figure 1.1 - Explain Plan of Task 1

TASK 2

Objectives

Create a Materialized View called SALES_CHAN_PROD_MV based on the query you wrote above for Question 1.Show the Materialized View being used by query rewrite by noting the Explain plan cost and table access.

Method

Use the keyword "**MATERIALIZED VIEW**" to Create the Materialized View with the name the SALES_CHAN_PROD_MV for Query 1 Explained above.

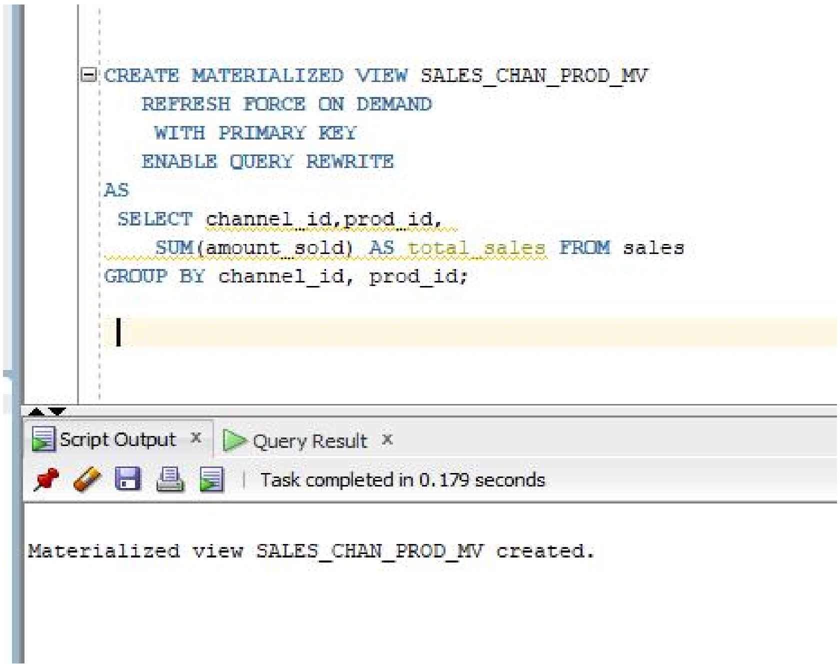
Query :

```
CREATE MATERIALIZED VIEW SALES_CHAN_PROD_MV

    REFRESH FORCE ON DEMAND

    WITH PRIMARY KEY

    ENABLE QUERY REWRITE

AS

SELECT channel_id,prod_id, SUM(amount_sold) AS total_sales FROM sales

GROUP BY channel_id, prod_id;
```

In this query:

- SALES_CHAN_PROD_MV is the name of the materialized view.
- The query within the 'AS' clause is the same query used in TASK 1.

OUTPUT



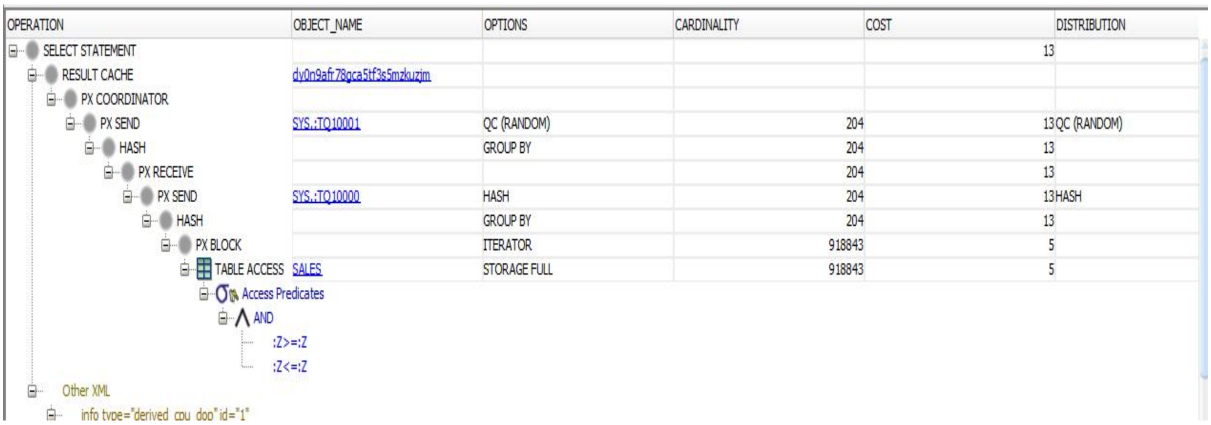
```
CREATE MATERIALIZED VIEW SALES_CHAN_PROD_MV
   REFRESH FORCE ON DEMAND
    WITH PRIMARY KEY
   ENABLE QUERY REWRITE
AS
 SELECT channel_id,prod_id,
    SUM(amount_sold) AS total_sales FROM sales
GROUP BY channel_id, prod_id;
```

```
Materialized view SALES_CHAN_PROD_MV created.
```

Figure 2 - Create Materialized view SALES_CHAN_PROD_MV

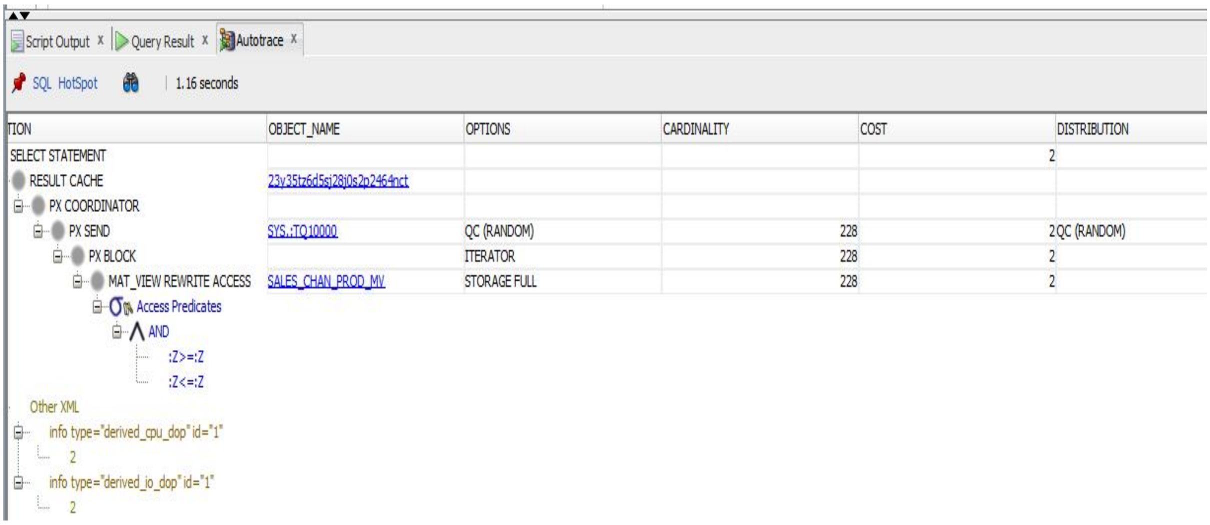
To show that the Materialized View is being used by query rewrite,you can use an SQL statement and request the Explain plan Option.

The Explain Plan shows that the query was rewritten to use the Materialized View SALES_CHAN_PROD_MV, which can lead to improved performance compared to querying the base table directly and also reduced the cost from 5 to 2,It's clear in below Figure 2.1 compared to Figure 2.2.In Figure 2.1 the table access is in sales table only but in Figure 2.2 its cleared mentioned that it access the SALES_CHAN_PROD_MV,which means that it have access to sales,products,channels tables.



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DISTRIBUTION
SELECT STATEMENT				13	
RESULT CACHE	dy0n9afr78gca5tf3s5mzkuzjm				
PX COORDINATOR					
PX SEND	SYS.:TQ10001	QC (RANDOM)	204	13	QC (RANDOM)
HASH		GROUP BY	204	13	
PX RECEIVE			204	13	
PX SEND	SYS.:TQ10000	HASH	204	13	HASH
HASH		GROUP BY	204	13	
PX BLOCK		ITERATOR	918843	5	
TABLE ACCESS	SALES	STORAGE FULL	918843	5	
Access Predicates					
AND					
:Z>=:Z					
:Z<=:Z					
Other XML					
info type="derived_cpu_dop" id="1"					

Figure 2.1 - Explain Plan of query 1 before materialized view creation



1.16 seconds

TION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DISTRIBUTION
SELECT STATEMENT				2	
RESULT CACHE	23y35tz6d5sj28j0s2p2464nct				
PX COORDINATOR					
PX SEND	SYS.:TQ10000	QC (RANDOM)	228	2	QC (RANDOM)
PX BLOCK		ITERATOR	228	2	
MAT_VIEW REWRITE ACCESS	SALES_CHAN_PROD_MV	STORAGE FULL	228	2	
Access Predicates					
AND					
:Z>=:Z					
:Z<=:Z					
Other XML					
info type="derived_cpu_dop" id="1"					
2					
info type="derived_io_dop" id="1"					
2					

Figure 2.2 - Explain Plan of SALES_CHAN_PROD_MV

TASK 3

Objectives

Write a query to return sales total grouped by channel description and product name. (No SQL extensions to GROUP BY required here). A sample of the output is as follows (225 rows):

Method

To return sales totals grouped by channel description and product name without using SQL extensions for grouping.The trick here is to use a CROSS connect to generate all conceivable combinations of channel descriptions and product names, and then connect the sales data to get the total sales for each combination.

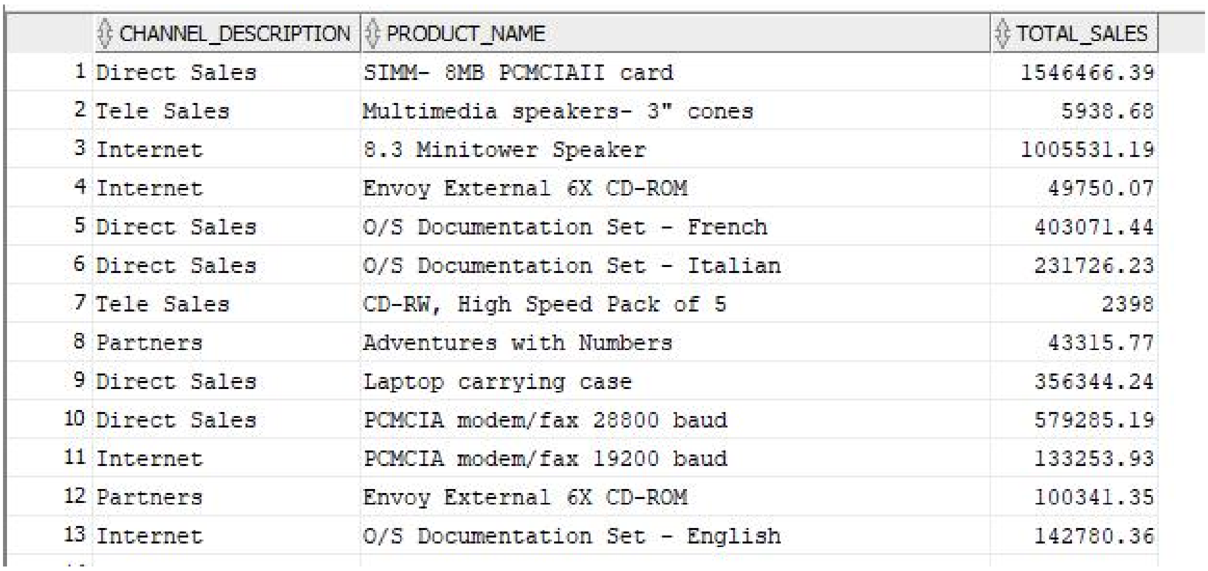
Query :

```
SELECT c.channel_desc  AS channel_description,
             p.prod_name AS product_name,
             SUM(s.amount_sold) AS total_sales
FROM  sales s
   JOIN channels c
           ON s.channel_id = c.channel_id
   JOIN products p
           ON s.prod_id = p.prod_id
GROUP BY c.channel_desc, p.prod_name;
```

In this query :

- Channels, Products, and Sales are the tables respectively include information about the channels, the products, and the sales.
- To extract the channel description and product name, we select the channel_desc column from the channels table and the prod_name column from the products table.
- To determine the overall sales amount for each combination of channel description and product name, we use the SUM() method.
- use a JOIN with the sales table to include sales data for those combinations that exist. If there are no sales data for a particular combination, the total_sales will be NULL for that combination.
- Finally, we are grouping the results by channel_desc and prod_name using the GROUP BY clause.
- This query will provide sales totals for all conceivable combinations, including those with no sales data, organized by channel description and product name. As mentioned in your sample output, the result should have 225 rows.

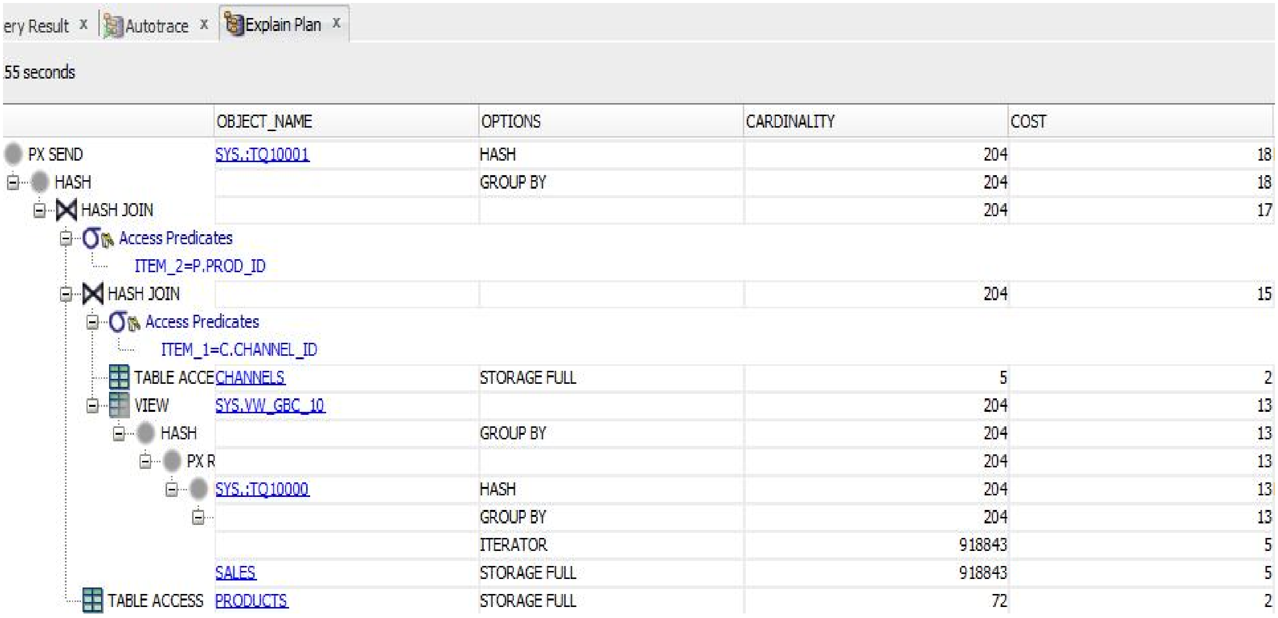
OUTPUT



	CHANNEL_DESCRIPTION	PRODUCT_NAME	TOTAL_SALES
1	Direct Sales	SIMM- 8MB PCMCIAII card	1546466.39
2	Tele Sales	Multimedia speakers- 3" cones	5938.68
3	Internet	8.3 Minitower Speaker	1005531.19
4	Internet	Envoy External 6X CD-ROM	49750.07
5	Direct Sales	O/S Documentation Set - French	403071.44
6	Direct Sales	O/S Documentation Set - Italian	231726.23
7	Tele Sales	CD-RW, High Speed Pack of 5	2398
8	Partners	Adventures with Numbers	43315.77
9	Direct Sales	Laptop carrying case	356344.24
10	Direct Sales	PCMCIA modem/fax 28800 baud	579285.19
11	Internet	PCMCIA modem/fax 19200 baud	133253.93
12	Partners	Envoy External 6X CD-ROM	100341.35
13	Internet	O/S Documentation Set - English	142780.36

Figure 3 - Sales total grouped by channel_desc and prod_name.

In Task 3 the execution plan cost is 2 for channels table,5 for sales table and 2 for products table and the query have access to the tables sales,products,channels.Its clearly mentioned in the figure 3.1.



	OBJECT_NAME	OPTIONS	CARDINALITY	COST
PX SEND	SYS.:TQ10001	HASH	204	18
HASH		GROUP BY	204	18
HASH JOIN			204	17
Access Predicates				
ITEM_2=P.PROD_ID				
HASH JOIN			204	15
Access Predicates				
ITEM_1=C.CHANNEL_ID				
TABLE ACCE	CHANNELS	STORAGE FULL	5	2
VIEW	SYS.VW_GBC_10		204	13
HASH		GROUP BY	204	13
PX R			204	13
	SYS.:TQ10000	HASH	204	13
		GROUP BY	204	13
		ITERATOR	918843	5
	SALES	STORAGE FULL	918843	5
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2

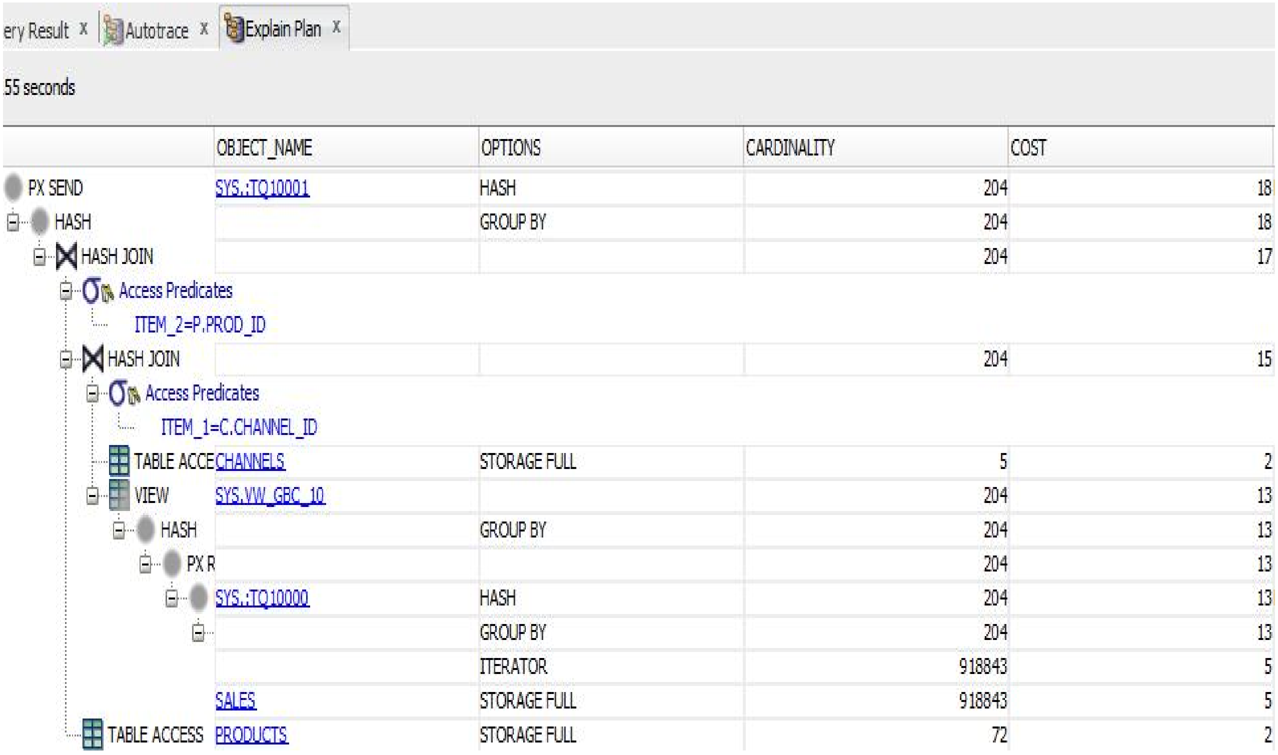
Figure 3.1 -Explain Plan of Task 3

TASK 4

Objectives

Does the Query written in Question 3 above use the Materialized View SALES_CHAN_PROD _MV?. If not explain why. Document your observations.

OUTPUT



55 seconds

	OBJECT_NAME	OPTIONS	CARDINALITY	COST
PX SEND	SYS.:TQ10001	HASH	204	18
HASH		GROUP BY	204	18
HASH JOIN			204	17
Access Predicates				
ITEM_2=P.PROD_ID				
HASH JOIN			204	15
Access Predicates				
ITEM_1=C.CHANNEL_ID				
TABLE ACCE	CHANNELS	STORAGE FULL	5	2
VIEW	SYS.VW_GBC_10		204	13
HASH		GROUP BY	204	13
PX R			204	13
	SYS.:TQ10000	HASH	204	13
		GROUP BY	204	13
		ITERATOR	918843	5
	SALES	STORAGE FULL	918843	5
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2

Figure 4 -Explain Plan of Task 3

RESULT

The query in Question 3 does not make use of the materialized view SALES _ CHAN _ PROD_MV. Instead, it directly connects the channels, products, and sales databases to compute the sales totals for each channel description and product name combination.It is clearly mentioned in Figure 6 that the tables are sales,channels and products and the execution plan cost is also indicated in the figure 4. The following are the reasons why the materialized view is not utilized in this query:

- TASK 3's query specifically refers to the basic tables channels, items, and sales. It makes no mention of the materialized perspective. Materialized views are designed to increase query efficiency by computerizing and storing results, but they must be explicitly mentioned in the query in order to be used.

- It is designed to create a cross-product of channel descriptions and product names and calculate the sales totals for these combinations, including those with no sales data. The materialized view, if it contains recomputed results for specific channel and product combinations, might not be a suitable fit for this particular query structure.

- Based on the main key associations between the basis tables' channels and products and the sales table, the JOIN conditions in the query are constructed. With or without a materialized view, these requirements enable the query to return sales information for every possible combination of product name and channel description.

- If the materialized view contains the precomputed sales totals for channel descriptions and product names, you could use it to potentially improve query performance, especially when dealing with large datasets. However, the query in Question 3 doesn't do this, so the materialized view remains unused.

TASK 5

Objectives

Create an appropriate Dimensional object to allow Q3 to use the Materialized View SALES_ CHAN_PROD_MV.

Method

Dimensions are usually stored in dimension tables.Need to create a dimension table for each dimension you want to define.For that use ‘CREATE DIMENSION’ Keyword.Within your dimension tables, define hierarchies by establishing parent-child relationships between attributes.Define levels for each attribute. Levels represent the granularity of data within the attribute.

Query :

```
CREATE DIMENSION sales_prod_dim
   LEVEL amount_sold IS ( sales.amount_sold )
   LEVEL product IS ( products.prod_id )
   LEVEL category IS ( products.prod_category )
   LEVEL prod_name IS ( products.prod_name )
   HIERARCHY spc_rollup (
       amount_sold   CHILD OF
       product       CHILD OF
       category      CHILD OF
       prod_name
   JOIN KEY (sales.prod_id) REFERENCES product )
  ATTRIBUTE category DETERMINES (
      products.prod_category,products.prod_category_desc,
      products.prod_subcategory, products.prod_subcategory_desc )
   ATTRIBUTE product DETERMINES (
      products.prod_desc, products.prod_weight_class,products.prod_unit_of_measure,
      products.prod_pack_size, products.prod_status,
      products.prod_list_price,products.prod_min_price  )
   ATTRIBUTE prod_name DETERMINES (
      products.prod_desc)
   ATTRIBUTE amount_sold DETERMINES (
```

sales.quantity_sold);

In this query :

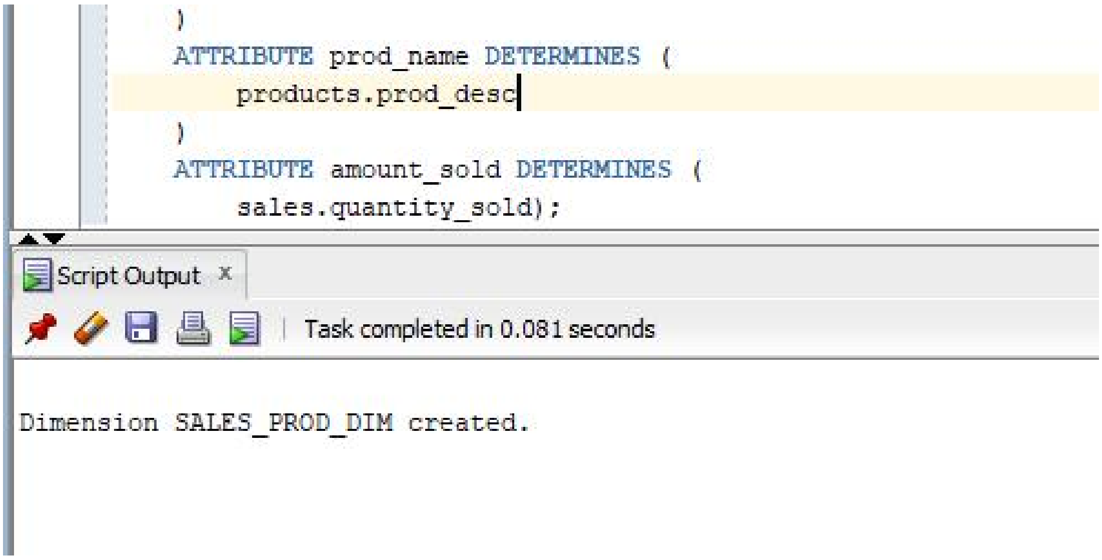
- sales_prod_dim is the dimension name
- The dimension is structured with several levels, which represent different attributes related to sales and products.
- A hierarchy named 'spc_rollup' is defined. Hierarchies define the parent-child relationships between levels.
- The hierarchy is defined with a join key relationship between levels.
- The query defines three attributes for this dimension:
  - ✓ For the category level, there are attributes such as

    category description, subcategory, and subcategory description.
  - ✓ For the product level, attributes include

    product description, weight class, unit of measure, pack size,

    status, list price, and minimum price.
  - ✓ The prod_name level only has an attribute for product description.
  - ✓ The amount_sold level has an attribute for quantity sold.

OUTPUT



```
)
ATTRIBUTE prod_name DETERMINES (
    products.prod_desc
)
ATTRIBUTE amount_sold DETERMINES (
    sales.quantity_sold);
```

Task completed in 0.081 seconds

```
Dimension SALES_PROD_DIM created.
```

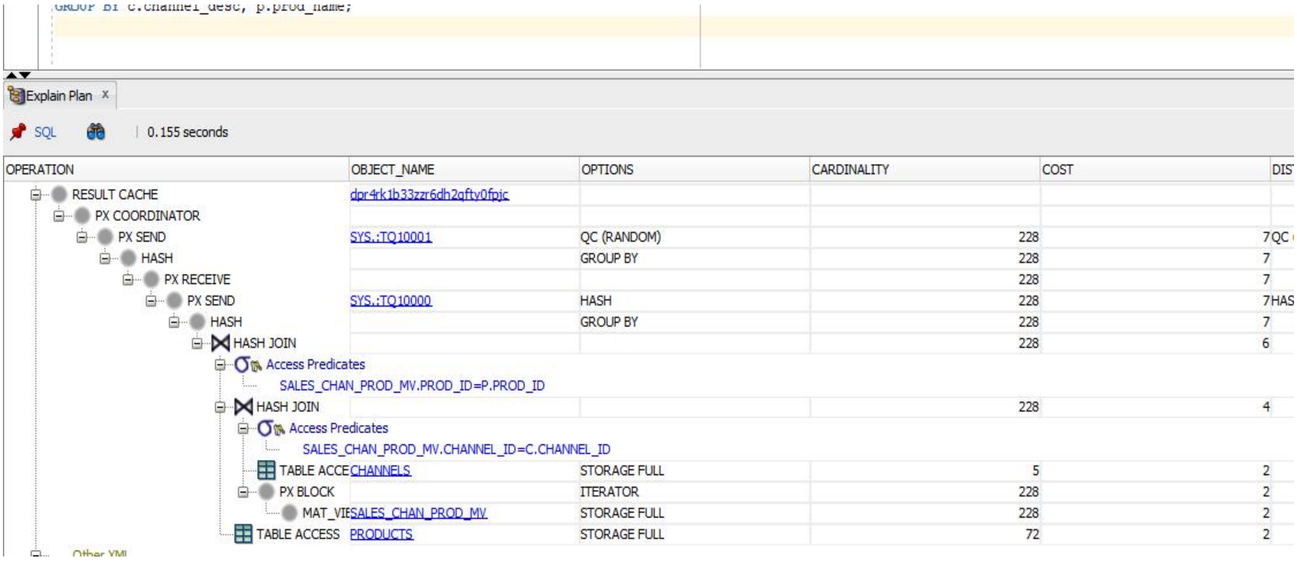
Figure 5- Dimension SALES_PROD_DIM Created

TASK 6

Objectives

Rerun Query 3. Does it use the materialized view SALES_CHAN_PROD_MV? What is the Explain plan cost and table access?

OUTPUT



```
GROUP BY c.channel_desc, p.prod_name;
```

Explain Plan

SQL | 0.155 seconds

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DIS
RESULT CACHE	dpr4rk1b33zzr6dh2qftv0fpjc				
PX COORDINATOR					
PX SEND	SYS.:TQ10001	QC (RANDOM)	228	7	QC
HASH		GROUP BY	228	7	
PX RECEIVE			228	7	
PX SEND	SYS.:TQ10000	HASH	228	7	HAS
HASH		GROUP BY	228	7	
HASH JOIN			228	6	
Access Predicates					
SALES_CHAN_PROD_MV.PROD_ID=P.PROD_ID					
HASH JOIN			228	4	
Access Predicates					
SALES_CHAN_PROD_MV.CHANNEL_ID=C.CHANNEL_ID					
TABLE ACCE	CHANNELS	STORAGE FULL	5	2	
PX BLOCK		ITERATOR	228	2	
MAT_VIE	SALES_CHAN_PROD_MV	STORAGE FULL	228	2	
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2	
Other XML					

Figure 6 -Explain Plan of task 3 after rerun

RESULT

The query in Question 3 use of the materialized view SALES_CHAN_PROD_MV when it is rerun after creating the dimension.Materialized views, also known as materialized tables or summary tables, are a critical component in data warehousing and database management systems. In figure 6 it is cleared mentioned that we are using SALES_CHAN_PROD_MV is used.Compared to figure 4 ,the execution plan cost is reduced and the query have access to the tables product ,channels and it also have table access to materialized table SALES_ CHAN_PROD_MV.It has no table access to the table sales in this Explain plan.Materialized View have several important advantages:

- Improved Query Performance.
- Reduced Overhead on Source Tables
- Support for Aggregations and Summaries
- Indexing and Optimization
- Reduced Network Latency
- Simplified Query Complexity
- Data Security

TASK 7

Objectives

Write a query to produce a report for management that require sales total grouped by channel description, product category and country name showing a total at each level of aggregation for France and Italy. Management do not want Peripherals and Accessories, Hardware or Photo product categories included in the report. Improve the readability of the report by using decode(29 records are returned as you see in the report in Figure 3).Experiment with ROLLUP, PARTIAL ROLLUP, CUBE, GROUPING SETS AND GROUPING. Document your results and observations.

Method

You can experiment with different SQL constructs such as ROLLUP, PARTIAL ROLLUP, CUBE, GROUPING SETS, and the DECODE function to create a report that summarize sales totals grouped by channel description, product category, and country name for France and Italy while excluding specific product categories such as Peripherals and Accessories, Hardware, and Photo.

Query

```
SELECT      ch.channel_desc as channel,
 DECODE(GROUPING(p.prod_category), 1, 'All Categories', p.prod_category) as Category,
 DECODE(GROUPING(shc.country_name), 1, 'France and Italy', shc.country_name) as
   Country ,
  SUM(amount_sold)as total_sales
   FROM       sales s, customers c, times t, channels ch, shcountries shc, products p
   WHERE       s.time_id = t.time_id
   AND          c.cust_id = s.cust_id
   AND          s.channel_id = ch.channel_id
   AND          s.prod_id=p.prod_id
   AND          p.prod_category IN ('Electronics', 'Software/Other')
   AND          ch.channel_desc IN ('Direct Sales', 'Internet')
   AND         shc.country_name IN ('France' , 'Italy')
   AND         t.calendar_month_desc IN ('2001-05','2001-06')
   GROUP BY ROLLUP
(ch.channel_desc,t.calendar_month_desc,shc.country_name,p.prod_category);
```

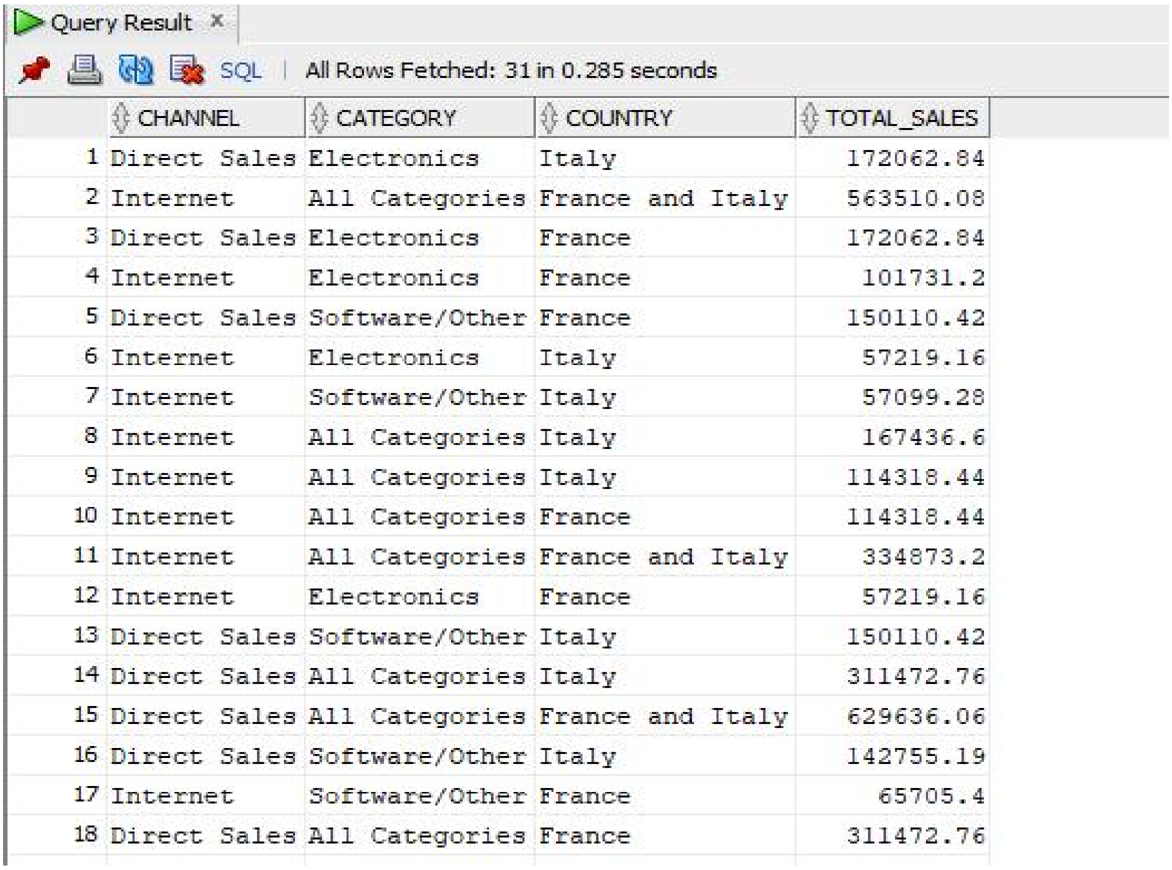
In this query:

- SELECT Statement select the specified columns and performs some calculations to get the desired results.Specified Columns are :
    - ✓ **channel_desc**: Represents the channel description where the sales occurred.
    - ✓ **Category**: Represents the product category.

- ✓ **Country**: Represents the country where the sales took place.
- ✓ **total_sales**: The total sales amount.

- ◆ FROM Clause: This part of the query lists the tables involved in the query. The tables are: sales, customers, times, channels, shcountries, and products.
- ◆ WHERE Clause: This is where conditions are specified to filter the data. The conditions are used to restrict the data to specific criteria.Here the conditions are
  - ✓ product category and country name showing a total at each level of aggregation for France and Italy.
  - ✓ Management do not want Peripherals and Accessories, Hardware or Photo product categories.
- ◆ COLUMN BY ROLLUP Clause: This clause is used to organise the information. By constructing several layers of aggregation, it creates a summary of sales data depending on the designated columns.
- ◆ The DECODE function is used to give some of the output's values more illuminating names. To make the output more understandable, for example, it substitutes '1' with 'All Categories' and '1' with 'France and Italy'.

OUTPUT



All Rows Fetched: 31 in 0.285 seconds

	CHANNEL	CATEGORY	COUNTRY	TOTAL_SALES
1	Direct Sales	Electronics	Italy	172062.84
2	Internet	All Categories	France and Italy	563510.08
3	Direct Sales	Electronics	France	172062.84
4	Internet	Electronics	France	101731.2
5	Direct Sales	Software/Other	France	150110.42
6	Internet	Electronics	Italy	57219.16
7	Internet	Software/Other	Italy	57099.28
8	Internet	All Categories	Italy	167436.6
9	Internet	All Categories	Italy	114318.44
10	Internet	All Categories	France	114318.44
11	Internet	All Categories	France and Italy	334873.2
12	Internet	Electronics	France	57219.16
13	Direct Sales	Software/Other	Italy	150110.42
14	Direct Sales	All Categories	Italy	311472.76
15	Direct Sales	All Categories	France and Italy	629636.06
16	Direct Sales	Software/Other	Italy	142755.19
17	Internet	Software/Other	France	65705.4
18	Direct Sales	All Categories	France	311472.76

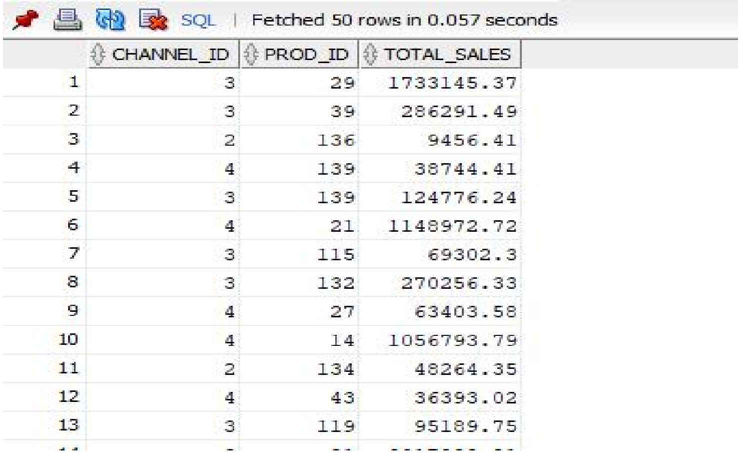
Figure 7 - Total Sales For France and Italy(excluding Category Hardware and Photo).

Conclusion

Throughout this process, we covered a variety of database administration and data analysis topics, with an emphasis on analytical SQL, data warehousing capabilities, and SQL functions. We started out by talking about how crucial SQL functions are for filtering, organizing, and preparing data. Filtering, aggregation, and transformation functions—all crucial for data analysis—are among the robust data manipulation features provided by SQL.After that, we explored the principles of data warehousing, which included employing dimensional objects and materialized views. In data warehousing setups, materialized views are essential for precomputing and storing summarized data, which improves query performance. Using dimensional objects like logs and indexes, searches may run more smoothly.After that We looked into the use of GROUP BY extensions and analytical SQL functions for data warehouse aggregation and summarization. Strong constructs for producing multidimensional summaries and enabling sophisticated data analysis include ROLLUP, CUBE, PARTIAL ROLLUP, GROUPING SETS, and GROUPING.All in all, this procedure covered a number of aspects of database administration, SQL operations, data warehousing, and analytical SQL—all essential for effectively managing and evaluating data. It emphasized how critical it is to streamline data operations, increase data accessibility, and offer insightful data to help organizations make well-informed decisions. When working with enormous datasets and intricate reporting requirements, data professionals and analysts need to possess these abilities and methods.

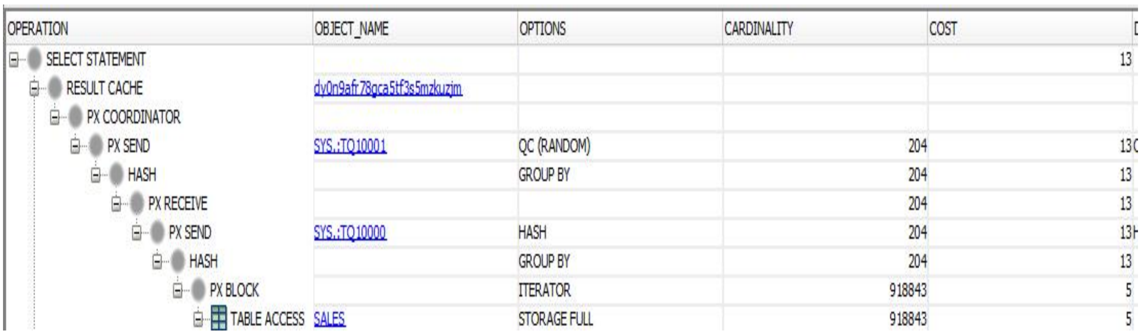
Appendices

Task 1



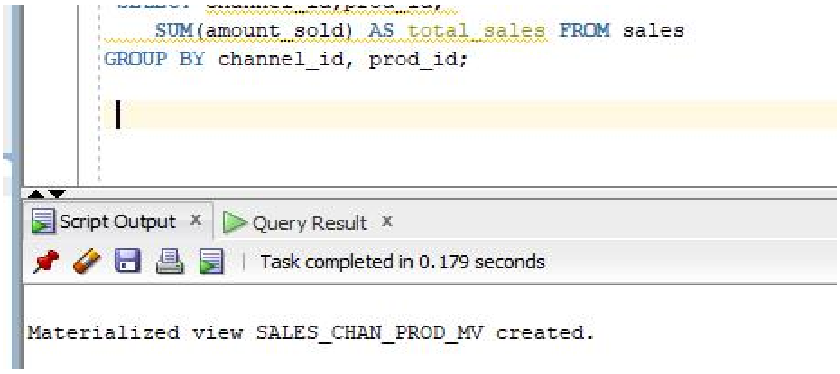
SQL | Fetched 50 rows in 0.057 seconds

	CHANNEL_ID	PROD_ID	TOTAL_SALES
1	3	29	1733145.37
2	3	39	286291.49
3	2	136	9456.41
4	4	139	38744.41
5	3	139	124776.24
6	4	21	1148972.72
7	3	115	69302.3
8	3	132	270256.33
9	4	27	63403.58
10	4	14	1056793.79
11	2	134	48264.35
12	4	43	36393.02
13	3	119	95189.75



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	D
SELECT STATEMENT				13	
RESULT CACHE	dy0n9afr78gca5tf3s5mzkuzjm				
PX COORDINATOR					
PX SEND	SYS.:TQ10001	QC (RANDOM)	204	13	C
HASH		GROUP BY	204	13	
PX RECEIVE			204	13	
PX SEND	SYS.:TQ10000	HASH	204	13	H
HASH		GROUP BY	204	13	
PX BLOCK		ITERATOR	918843	5	
TABLE ACCESS	SALES	STORAGE FULL	918843	5	

Task 2

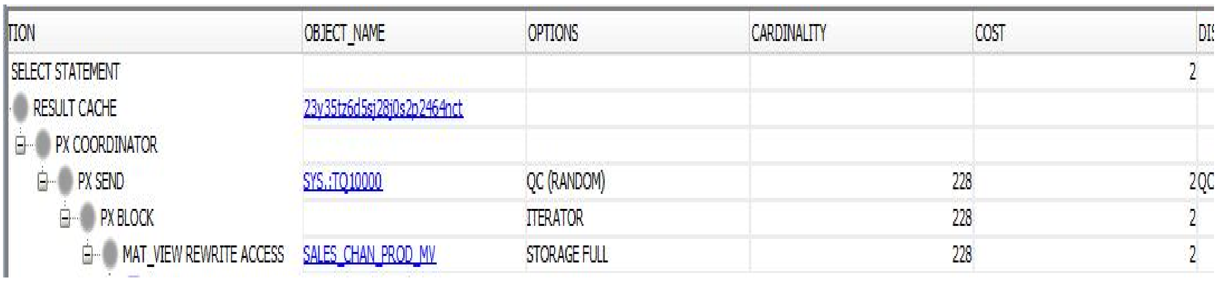


```
    SUM(amount_sold) AS total_sales FROM sales
GROUP BY channel_id, prod_id;
```

Script Output × | Query Result ×

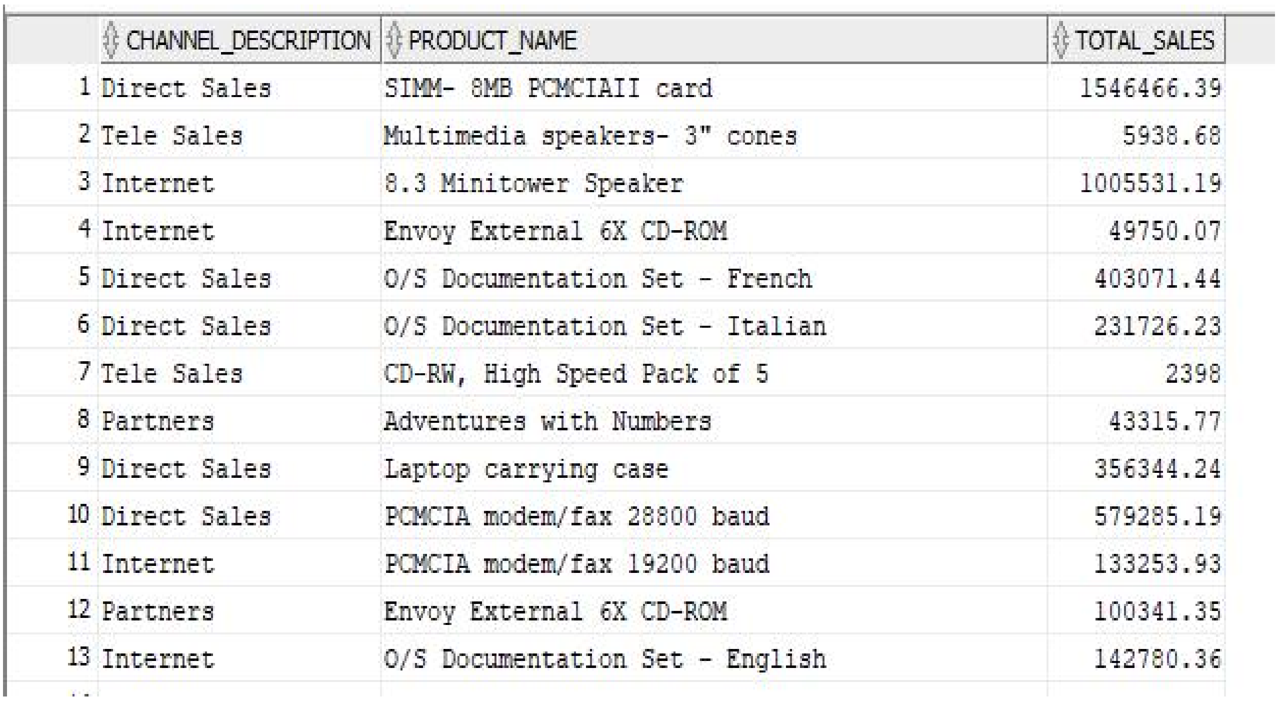
| Task completed in 0.179 seconds

```
Materialized view SALES_CHAN_PROD_MV created.
```

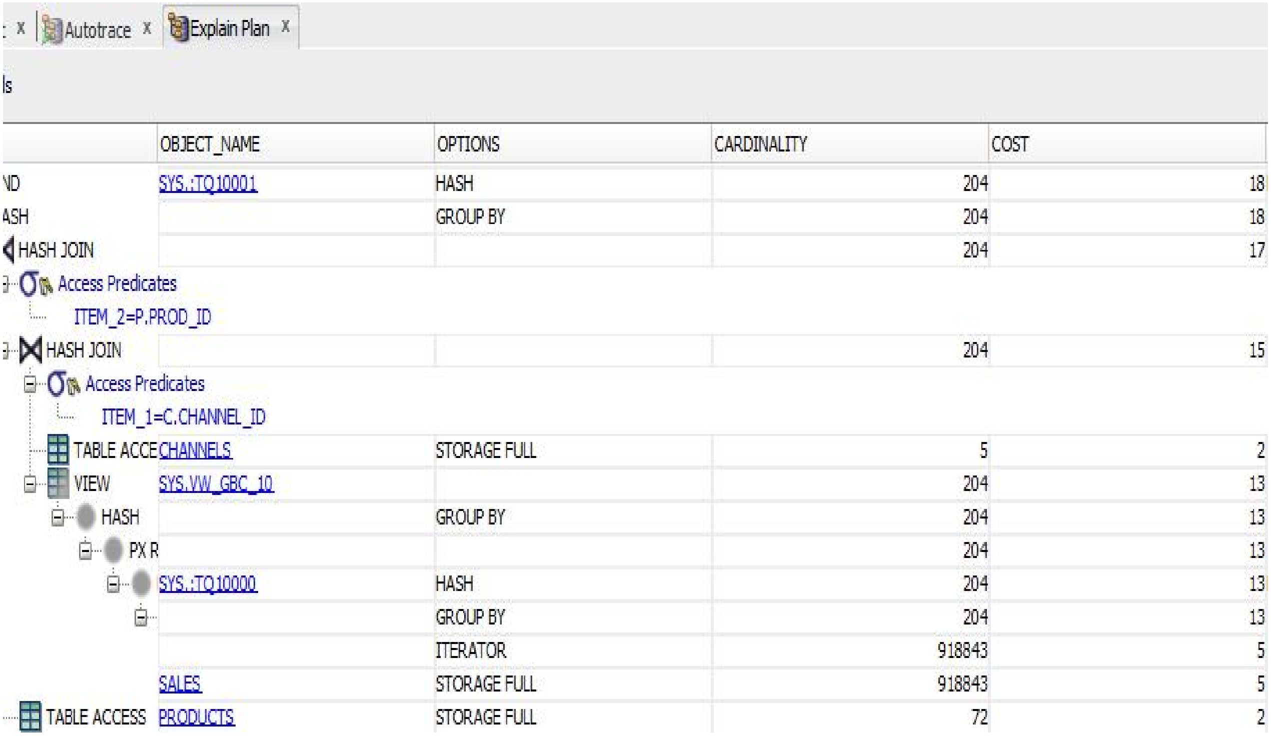


TION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DI
SELECT STATEMENT				2	
RESULT CACHE	23y35tz6d5sj28j0s2p2464nct				
PX COORDINATOR					
PX SEND	SYS.:TQ10000	QC (RANDOM)	228	2	QC
PX BLOCK		ITERATOR	228	2	
MAT_VIEW REWRITE ACCESS	SALES_CHAN_PROD_MV	STORAGE FULL	228	2	

Task 3

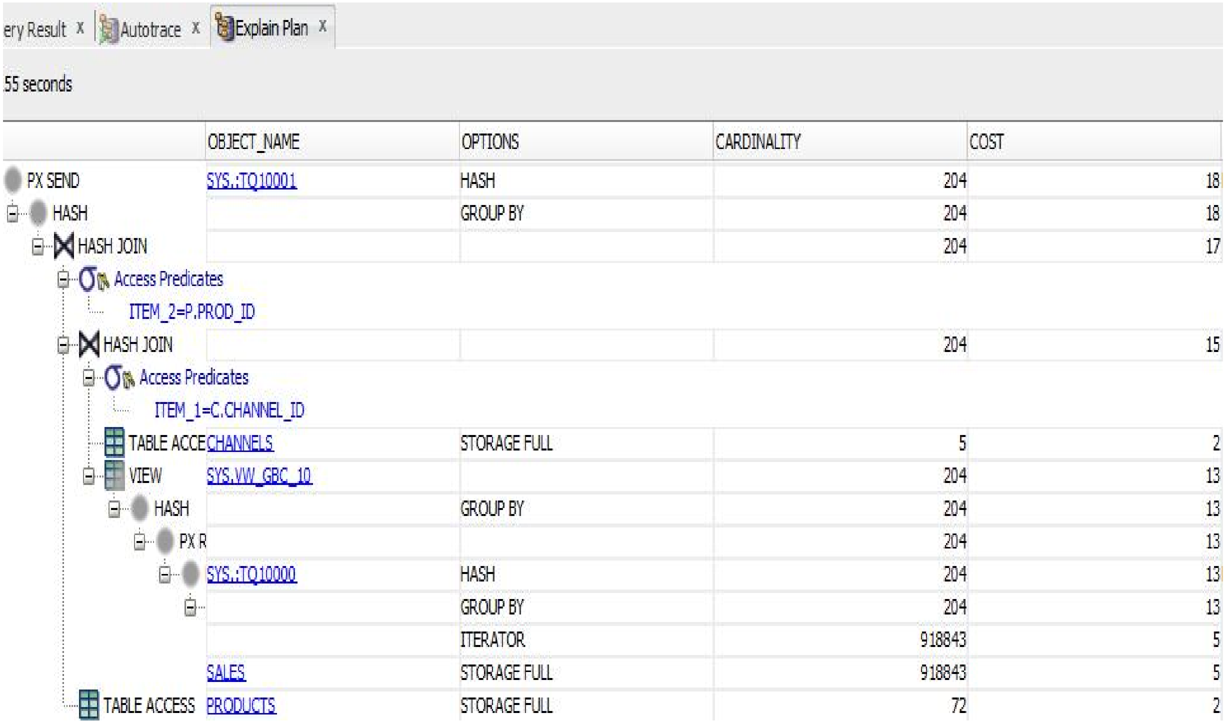


	CHANNEL_DESCRIPTION	PRODUCT_NAME	TOTAL_SALES
1	Direct Sales	SIMM- 8MB PCMCIAII card	1546466.39
2	Tele Sales	Multimedia speakers- 3" cones	5938.68
3	Internet	8.3 Minitower Speaker	1005531.19
4	Internet	Envoy External 6X CD-ROM	49750.07
5	Direct Sales	O/S Documentation Set - French	403071.44
6	Direct Sales	O/S Documentation Set - Italian	231726.23
7	Tele Sales	CD-RW, High Speed Pack of 5	2398
8	Partners	Adventures with Numbers	43315.77
9	Direct Sales	Laptop carrying case	356344.24
10	Direct Sales	PCMCIA modem/fax 28800 baud	579285.19
11	Internet	PCMCIA modem/fax 19200 baud	133253.93
12	Partners	Envoy External 6X CD-ROM	100341.35
13	Internet	O/S Documentation Set - English	142780.36



	OBJECT_NAME	OPTIONS	CARDINALITY	COST
ND	SYS.:TQ10001	HASH	204	18
ASH		GROUP BY	204	18
HASH JOIN			204	17
Access Predicates				
ITEM_2=P.PROD_ID				
HASH JOIN			204	15
Access Predicates				
ITEM_1=C.CHANNEL_ID				
TABLE ACCE	CHANNELS	STORAGE FULL	5	2
VIEW	SYS.VW_GBC_10		204	13
HASH		GROUP BY	204	13
PX R			204	13
	SYS.:TQ10000	HASH	204	13
		GROUP BY	204	13
		ITERATOR	918843	5
	SALES	STORAGE FULL	918843	5
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2

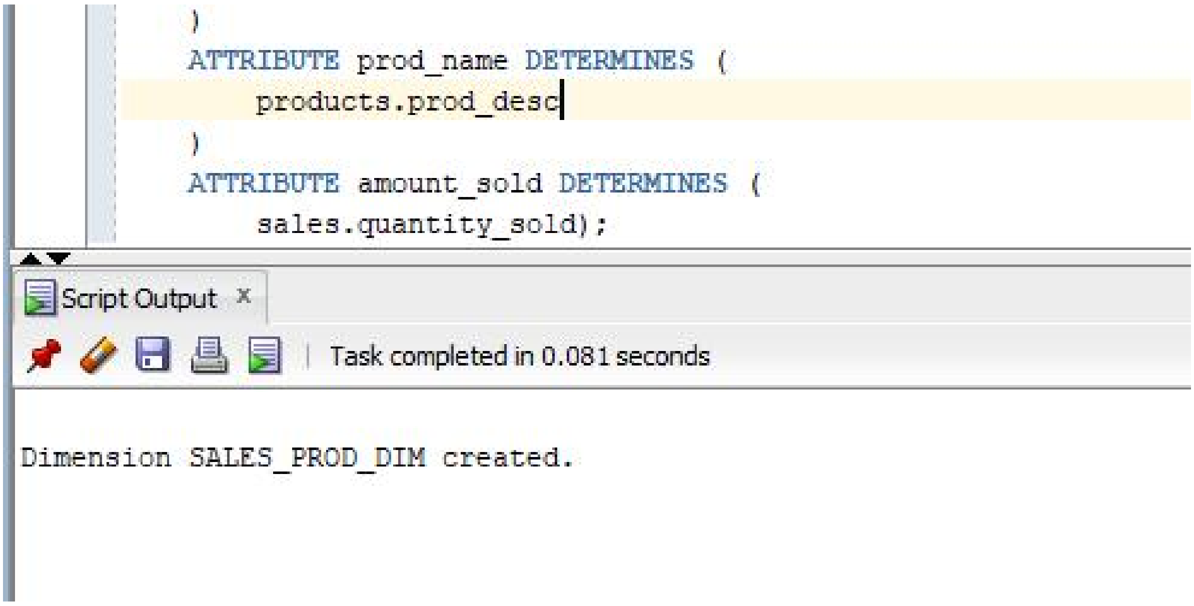
TASK 4



55 seconds

	OBJECT_NAME	OPTIONS	CARDINALITY	COST
PX SEND	SYS.:TQ10001	HASH	204	18
HASH		GROUP BY	204	18
HASH JOIN			204	17
Access Predicates				
ITEM_2=P.PROD_ID				
HASH JOIN			204	15
Access Predicates				
ITEM_1=C.CHANNEL_ID				
TABLE ACCE	CHANNELS	STORAGE FULL	5	2
VIEW	SYS.VW_GBC_10		204	13
HASH		GROUP BY	204	13
PX R			204	13
	SYS.:TQ10000	HASH	204	13
		GROUP BY	204	13
		ITERATOR	918843	5
	SALES	STORAGE FULL	918843	5
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2

TASK 5

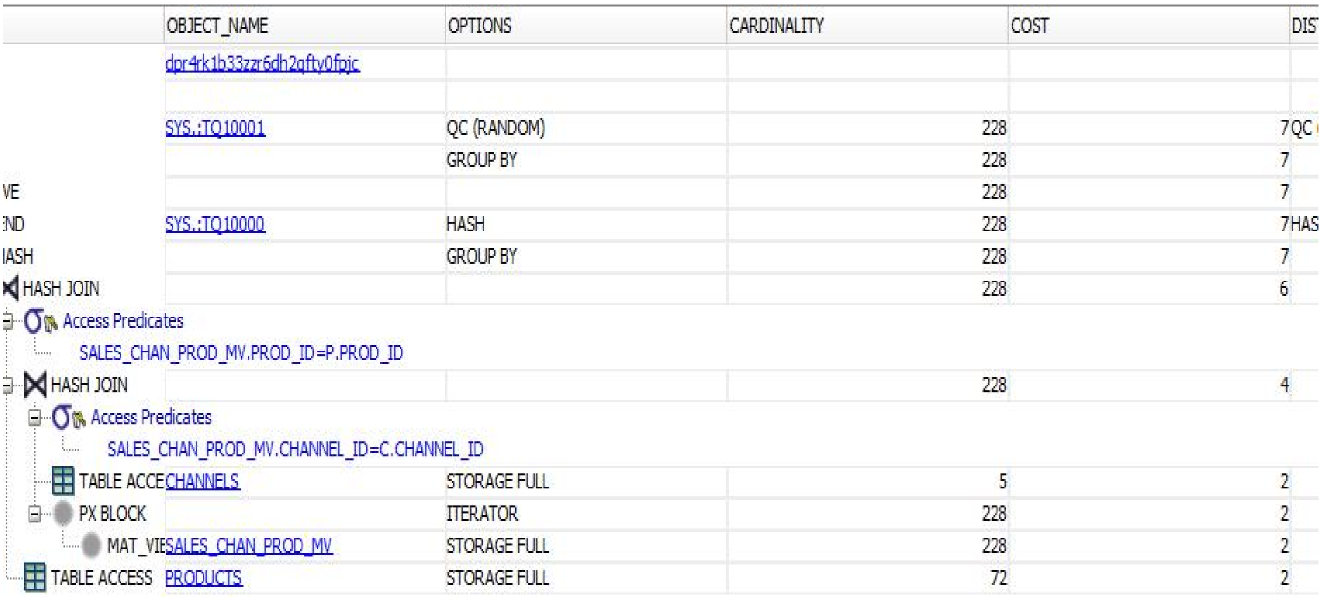


```
)
ATTRIBUTE prod_name DETERMINES (
    products.prod_desc
)
ATTRIBUTE amount_sold DETERMINES (
    sales.quantity_sold);
```

Task completed in 0.081 seconds

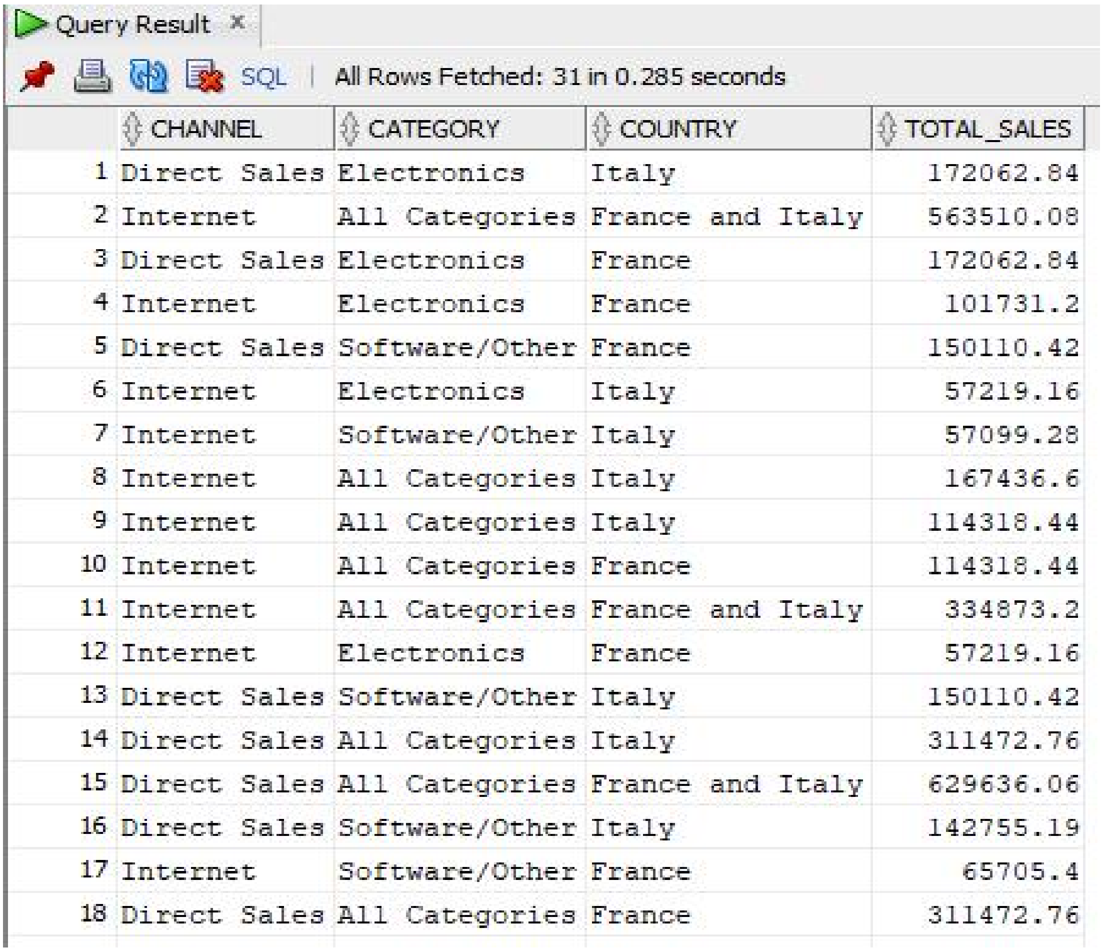
```
Dimension SALES_PROD_DIM created.
```

Task 6



	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DIS
	dpr4rk1b33zzr6dh2qftv0fpjc				
	SYS.:TQ10001	QC (RANDOM)	228	7	QC
		GROUP BY	228	7	
VE			228	7	
ND	SYS.:TQ10000	HASH	228	7	HAS
IASH		GROUP BY	228	7	
HASH JOIN			228	6	
Access Predicates					
SALES_CHAN_PROD_MV.PROD_ID=P.PROD_ID					
HASH JOIN			228	4	
Access Predicates					
SALES_CHAN_PROD_MV.CHANNEL_ID=C.CHANNEL_ID					
TABLE ACCE	CHANNELS	STORAGE FULL	5	2	
PX BLOCK		ITERATOR	228	2	
MAT_VIE	SALES_CHAN_PROD_MV	STORAGE FULL	228	2	
TABLE ACCESS	PRODUCTS	STORAGE FULL	72	2	

TASK 7



All Rows Fetched: 31 in 0.285 seconds

	CHANNEL	CATEGORY	COUNTRY	TOTAL_SALES
1	Direct Sales	Electronics	Italy	172062.84
2	Internet	All Categories	France and Italy	563510.08
3	Direct Sales	Electronics	France	172062.84
4	Internet	Electronics	France	101731.2
5	Direct Sales	Software/Other	France	150110.42
6	Internet	Electronics	Italy	57219.16
7	Internet	Software/Other	Italy	57099.28
8	Internet	All Categories	Italy	167436.6
9	Internet	All Categories	Italy	114318.44
10	Internet	All Categories	France	114318.44
11	Internet	All Categories	France and Italy	334873.2
12	Internet	Electronics	France	57219.16
13	Direct Sales	Software/Other	Italy	150110.42
14	Direct Sales	All Categories	Italy	311472.76
15	Direct Sales	All Categories	France and Italy	629636.06
16	Direct Sales	Software/Other	Italy	142755.19
17	Internet	Software/Other	France	65705.4
18	Direct Sales	All Categories	France	311472.76